ISD Checklist

Number: 580-CK-005-01
Effective Date: 03/01/2004
Expiration Date: 03/01/2009

Approved By: (signature)
Name: Joe Hennessy
Title: Chief, ISD

Responsible Office: 580/Information Systems Division (ISD)
Title: Contents of the Software Requirements Review

Asset Type: Checklist
PAL Number: 2.2.1.6

Contents of the Software Requirements Review (SRR)

At the SRR, the development team presents the detailed requirements and specifications for the software system or subsystem to independent experts, management, customers, users and other interested parties. The purpose of the SRR is to verify that these software requirements are clear, complete, consistent, feasible, traceable, and testable.

Use the following checklist to ensure that the SRR addresses all the information necessary to allow review participants to make these determinations. Successful completion of the SRR certifies that the requirements form a satisfactory basis for proceeding with software design.

❑ **Introduction**

- ❑ Goals of this review; review prerequisites, scope, agenda, and products
- ❑ Introduction to the review panel and review process to be followed (e.g., Request for Action (RFA) or Review Item Disposition (RID) reports, schedule)

Project Overview

- ❑ Purpose and overview of the software project or overall Mission Project (e.g., Mission goals, orbit, launch, operational characteristics)
- ❑ Software project organization and key personnel, including how the project fits within the overall Mission organization and identification of the project's systems engineers at the Mission Project level
- ❑ External dependencies

Software Development Overview

- ❑ Development methodology being followed, focusing on requirements definition, requirements analysis, and high-level design
- ❑ Identification of all source documents from which software requirements were derived
- ❑ Explanation of any prototypes, simulations, or other studies of requirements feasibility

❑ **Requirements context** – Overview of system concepts and requirements from which the detailed requirements and specifications were derived

Hardware Context

- ❑ Diagrams showing the hardware environment in which the software will operate (e.g., processors, buses, sensors, communications networks)
- ❑ Major software functional elements, showing data flows between software and hardware elements
- ❑ Identification of any resource limitations or allocations associated with the hardware elements on which software depends, including processors, memory, data storage
- ❑ Identification of Interface Requirements Documents (IRDs) and Interface Control Documents (ICDs) and the organizations developing or acquiring hardware elements; IRD and ICD status and need dates for any open or TBD issues

Software Context

- ❑ Diagrams showing external software interfaces to this software system/subsystem.
- ❑ Identification of software IRDs and ICDs and the development organizations associated with relevant external software elements; IRD and ICD status and need dates for open/TBD issues
- ❑ Architectural diagrams showing any COTS, GOTS, heritage software, operating systems, and/or development languages specified in the higher-level requirements or system concept documents

Contents of the Software Requirements Review (SRR) *(Continued)*

Operations Context

- ❑ Diagram of operations characteristics (e.g., ground contacts, science operations)
- ❑ Operational scenarios for the major functional areas, including input mechanisms, processing, output, interfaces, and identification of users
- ❑ Software control modes and mode transition rules
- ❑ Fault detection, isolation and recovery (FDIR) strategies
- ❑ Safety hazard reduction strategies
- ❑ Security features

❑ **Functional requirements –** Address the following *for **each major functional area*** of the software:

- ❑ Definition of high-level requirements for each functional area
- ❑ Block diagram of the major software components in each functional area, their interfaces and data flows
- ❑ Definition of relevant operational modes (e.g., nominal, critical, contingency)
- ❑ Critical and/or controversial requirements, including open issues and areas of concern
- ❑ Requirements needing clarification or additional information
- ❑ Trace of each requirement to the higher-level (e.g., mission) document from which it was derived

❑ **Performance requirements**

- ❑ Identification of performance requirements for the software (e.g., system response times, failure recovery times, output data availability)
- ❑ Description of critical timing relationships and constraints (e.g., external hardware interface timing, command execution timing)

❑ **Design strategy**

- ❑ Explanation of design drivers and design decisions that have been made, including software architecture, operating systems, reuse of existing software, and selection of COTS components
- ❑ Resource goals and preliminary sizing estimates in the context of available hardware allocations; strategies for measuring and tracking resource utilization
- ❑ Initial Build Plan

❑ **Qualification requirements**

- ❑ Discussion of overall software test strategy, including the test levels (unit, integration, build, and system-level testing), test types (interface, load/stress, regression), and test tools
- ❑ Software development and test environments, including processors, operating systems, communications equipment, simulators and their fidelity
- ❑ Traceability of requirements to build and system-level tests
- ❑ Methodology for verifying the system requirements and acceptance criteria
- ❑ Test tool requirements and development plans

❑ **Software Management Plan**

Management approach

- ❑ Software team(s) organization, WBS, and reporting relationships
- ❑ Software size estimates, budgets, and staffing
- ❑ Development schedules showing key receivables, deliverables, and dependencies
- ❑ Management overview, including metrics to be collected
- ❑ Risks and risk mitigation plans
- ❑ Infusion of previous Lessons Learned; collection of new Lessons Learned

Technical approach

- ❑ Requirements management approach and tools
- ❑ Configuration Management, Product Assurance, and Software Safety approaches and tools
- ❑ Software development environment(s) and tools
- ❑ Documentation summary and schedule

❑ **Status –** Current schedule, milestone, and cost/effort status

❑ **Issues, TBDs, and action items**